Combining Word Embeddings with Bilingual Orthography Embeddings for Bilingual Dictionary Induction

Silvia Severini, Viktor Hangya, Alexander Fraser, Hinrich Schütze
Center for Information and Language Processing
LMU Munich, Germany
{silvia, hangyav, fraser}@cis.uni-muenchen.de

Abstract

Bilingual dictionary induction (BDI) is the task of accurately translating words to the target language. It is of great importance in many low-resource scenarios where cross-lingual training data is not available. To perform BDI, bilingual word embeddings (BWEs) are often used due to their low bilingual training signal requirements. They achieve high performance, but problematic cases still remain, such as the translation of rare words or named entities, which often need to be transliterated. In this paper, we enrich BWE-based BDI with transliteration information by using *Bilingual Orthography Embeddings* (BOEs). BOEs represent source and target language transliteration word pairs with similar vectors. A key problem in our BDI setup is to decide which information source – BWEs (or semantics) vs. BOEs (or orthography) – is more reliable for a particular word pair. We propose a novel classification-based BDI system that uses BWEs, BOEs and a number of other features to make this decision. We test our system on English-Russian BDI and show improved performance. In addition, we show the effectiveness of our BOEs by successfully using them for transliteration mining based on cosine similarity.

1 Introduction

The task of Bilingual Dictionary Induction is defined as finding target language translations of source language words. It is an important building block in the area of Machine Translation (MT) and it is one of the main tasks for bilingual word embedding evaluation (Mikolov et al., 2013b; Vulic and Korhonen, 2016). Recent work shows that good performance can be achieved relying only on BWEs, which can be built with only a weak bilingual signal, such as a small seed lexicon of a few thousand word pairs (Mikolov et al., 2013b) or common tokens in the source and target languages (Artetxe et al., 2017). In addition, they can even be built without any bilingual signal (Conneau et al., 2018; Artetxe et al., 2018), making them the basis of unsupervised MT systems (Lample et al., 2018; Artetxe et al., 2019).

Standard BDI learns word representations based on approaches that exploit solely word-level information such as *word2vec* (Mikolov et al., 2013a) or *fasttext* (Bojanowski et al., 2017) and then map them to a shared BWE space. Although BWE-based approaches show high BDI performance, they struggle with a subset of hard-to-translate words such as named entities for which orthographic information should be used instead of semantic information. Several approaches have integrated orthographic information into the BDI system. Heyman et al. (2017) relied on character-level information in their classification based BDI system by using an RNN architecture. Braune et al. (2018) combined orthographic information with BWE-based word similarity information using an ensembling method. Both of these approaches showed improved results, but they relied on Levenshtein distance to get translation candidates for a source word during prediction, which is not applicable for language pairs with different scripts. To bridge the gap between languages with different scripts, a transliteration system was employed by Severini et al. (2020). They followed the approach of Braune et al. (2018) but used the transliteration system instead of Levenshtein distance to get candidates used in the ensembling model. On the other hand, as they also showed,

Proceedings of the 28th International Conference on Computational Linguistics, pages 6044–6055
Barcelona, Spain (Online), December 8-13, 2020

the ensembling approach often fails to decide correctly if a word has to be transliterated or not; this is because there are only two independent scores available, the score of the (semantic) BWE, and the score of the (orthographic) transliteration model.

In this paper, we present our novel approach to BDI focusing on words that have to be transliterated to another script, which is especially important for low-frequency words, but also relevant for high-frequency named entities. Our aim is to improve BDI systems in two respects: (i) eliminating the need for language specific orthographic information, such as is used in Levenshtein distance, and (ii) to be able to better decide when to choose transliteration over semantic translation. We propose a new approach for language pairs with different scripts by combining semantic information with orthographic information. For the latter we introduce *Bilingual Orthographic Embeddings* (BOEs) of words, which represent transliteration pairs in the source and target language with similar vectors. We build BOEs using a novel transliteration system trained jointly for both language directions. We refer to this novel system as *seq2seqTr*. *seq2seqTr* can also be used to extract candidate transliterations for a source word. It is applicable to any language pair (as opposed to Levenshtein distance). To make a more informed decision about which words should be transliterated (which means we shoud primarily trust the BOEs) and which should be semantically translated (which means we should primarily trust the BWEs), we use a classification approach similar to (Heyman et al., 2017), exploiting our pretrained encoder from seq2seqTr. In contrast to their approach, we use additional features, such as frequency, length, similarity scores, and the ranks assigned by the semantic and character-level submodels, and show that they are necessary to make the right decision.

We test our system on the *English-Russian* (En-Ru) data provided in the BUCC 2020 shared task (Rapp et al., 2020). Test dictionaries were released in three frequency categories: high, middle and low. We evaluate our system on all three sets, both separately and jointly, and show improved performance on all three frequency ranges compared with previous approaches. Furthermore, we show that our classification system is more robust than the ensembling of Severini et al. (2020), which required specialized tuning on each frequency set. Lastly, we conduct a further analysis of the quality of the proposed BOEs by running transliteration mining on the NEWS 2010 shared task data (Kumaran et al., 2010) by using the vector similarity of Bilingual Orthographic Embeddings of words. We show good performance on the task indicating the usefulness of BOEs for other downstream tasks.

2 Related Work

2.1 Bilingual Dictionary Induction

BWEs are often used for solving BDI tasks by calculating cosine similarity of word pairs and taking the n most similar target candidates as translations for a source word. As opposed to general MT based approaches that rely on parallel sentences, BWEs are also effective when only a small seed lexicon is provided (e.g., (Mikolov et al., 2013b)). Conneau et al. (2018) and Artetxe et al. (2018) dispense with seed dictionaries and iteratively improve the mapping from an initial weak solution in a self-learning approach. This setting provides a building block for unsupervised MT and is particularly effective in the low-resource setting where less parallel seed data is available (Artetxe et al., 2019; Lample et al., 2018).

BWE-based methods perform worse for low frequency words due to poor vector representations (Braune et al., 2018; Riley and Gildea, 2018; Czarnowska et al., 2019). Koehn and Knight (2002) and Haghighi et al. (2008) show that orthographic features help in the translation process. Languages with a common alphabet (e.g., English/German) often have word pairs with similar orthography (e.g., *Concepts/Konzepte*, *Philosophies/Philosophien*), especially in the case of low frequency words. Riley and Gildea (2018) integrate orthographic information into the vector representation of such words and into the mapping procedure of BWEs to improve their quality. Braune et al. (2018) use character n-gram representations and Levenshtein distance to improve BDI while Heyman et al. (2017) extract this feature automatically from training data. In languages with different scripts (e.g., English/Russian), the source word is often written with the closest corresponding letters of the target alphabet, i.e., it is transliterated. *Richard/*Ричард and *integrator/*интегратор are examples of transliterations between English and Russian. Irvine and Callison-Burch (2017) applied Levenshtein distance to language pairs with different

alphabets by first transliterating from non-Latin to Latin scripts. In contrast, we use a novel transliteration model that encodes the relevant information directly into BOEs without requiring a separate transliteration step.

2.2 Transliteration

As motivated above, transliteration mining is an important task that bridges the gap between languages with different scripts. The NEWS transliteration shared task has been a continuous effort to promote research on this task since 2009 (Li et al., 2009; Chen et al., 2018). A fully unsupervised transliteration model was proposed by Sajjad et al. (2017); it consists of interpolated statistical sub-models for transliteration and non-transliteration detection. In this work we follow a similar idea and propose an unsupervised neural network based system to look for transliteration pairs of source words as possible translation candidates. We also use this system to build BOEs of words.

In a parallel sentence mining approach, Artetxe and Schwenk (2019) use a shared encoder and decoder for all languages to build a language agnostic sentence encoder. They use the encoder representations as sentence embeddings to efficiently mine parallel sentences. Similarly, our BOEs are extracted from a single language agnostic encoder, for both English and Russian. In an ablation study, we check the quality of our BOEs on the NEWS 2010 shared task (Kumaran et al., 2010); see below.

3 Approach

To tackle the BDI task we exploit BWEs, character-level information (in the form of BOEs) and manually engineered features – such as word frequency and length – and integrate them into a classifier that predicts if the word pair is a translation or not. We extract candidate translations for a source word based on (i) BWEs and (ii) *seq2seqTr* (our transliteration model) as described in the following sections. Finally, we rerank the two groups of candidates with our classification system described below and take the top ranked candidates as our prediction.

3.1 Bilingual Word Embeddings

To create BWEs we use monolingual word embeddings (MWEs), learned with *fasttext skipgram* (Bojanowski et al., 2017), and align them to a shared space with a seed dictionary that consists of high frequency word pairs using the approach of Artetxe et al. (2018). We then generate translation candidates for each source word by taking the n target language words that are the most similar. Our similarity measure is the cosine-similarity-based *Cross-Domain Similarity Local Scaling* (CSLS) measure (Conneau et al., 2018). We use these target candidates along with the corresponding source words as classification samples during prediction.

3.2 The Transliteration Model: seq2seqTr

In this work, we focus on two languages that have different alphabets since our aim is to improve application scenarios in which the Levenshtein distance is not applicable. To address language pairs with different scripts and pairs of infrequent words such as named entities, we propose seq2seqTr, a novel transliteration system. seq2seqTr is trained on a list of word pairs that are translations of one another – both transliterations and non-transliterations. It is unsupervised because we do not rely on labels to distinguish between transliteration and non-transliteration training pairs. seq2seqTr is a character-level sequence-to-sequence model (Sutskever et al., 2014) with a single-layer encoder and a single-layer decoder. The encoder is a bidirectional GRU (Cho et al., 2014) while the decoder is unidirectional with attention (Luong et al., 2015). The input characters are represented as vectors. Figure 1 (bottom) depicts the model. We use this model to calculate the probability of each word in the target language vocabulary with respect to each source test word. The probability corresponds to the average negative log likelihood of the characters in the target word with respect to the source word. We select n target transliteration candidates for each source word.

To train seq2seqTr, we start with the same training dictionary as for building BWEs. Since it contains many non-transliteration pairs, we reduce their number with an iterative cleaning process. The dictionary

is considered "cleaner" if it contains fewer non-transliteration pairs than the initial one; but it should still have enough transliteration pairs to allow the effective training of the transliteration model. In more detail, we first select 100 pairs randomly from the training dictionary. We call this the *comparison set*. We limit the sampling to pairs that have the same length to make sure that a sufficient number of transliterations is in the comparison set. Then, we split the dictionary[1] into train and test (80%-20%) and we train seq2seqTr on this new training set with early stopping. We calculate the 95% confidence interval for the scores (average log likelihood, see above) of the comparison set. Let θ be the upper bound of the confidence interval. We then remove the pairs in the secondary test set that have a score lower than θ. Finally, we merge secondary training set and cleaned secondary test set, shuffle them and iterate the process until we can no longer remove pairs. The intuition for the choice of θ is that we only want to keep pairs in the final training set that are transliterations with high probability – pairs whose scores are clearly higher than the typical scores in the comparison set (i.e., are larger than θ) should have this property. The same iterative method is run to clean the development set portion of the dictionary.

3.3 Bilingual Orthography Embeddings

As a representation that is informative about orthography of source and target words, we build Bilingual Orthography Embeddings (BOEs). The BOE space is a common representation space – just like the BWE space – and transliteration pairs are represented with similar vectors. We again use the same training set (the cleaned training set) and our seq2seqTr architecture, but we tune GRU hidden and character embedding sizes for the BDI task. More importantly, we employ a slightly different training procedure to tie the two languages together and to build a language agnostic encoder that works for both source and target languages. To this end, we train seq2seqTr with source-to-target and target-to-source word pairs where we indicate the output language using a language specific marker at the first position of the target decoder output. Since we want a language agnostic encoder, we do not use such a marker on the encoder side.

In addition to source-to-target and target-to-source word pairs, we also train seq2seqTr on source-to-source and target-to-target pairs, i.e., we also train the model as an autoencoder. We use the same words for within-language as for cross-lingual pairs. Without training seq2seqTr to be an autoencoder, the encoder representations for Russian and English would not be in the same subspace. As the BOE representation of a word, we take the final hidden state of the encoder GRU layer since it is also used to initialize the decoder of the complete transliteration model. We also experimented with taking the averaged output states of the final encoder layer as BOEs and decoder initialization, similar to (Artetxe and Schwenk, 2019), but it gave slightly worse results.

3.4 BDI Systems

We first introduce our baselines and then explain our classification model.

Ensembling Baseline Severini et al. (2020) used ensembling to combine the candidate translations for a source word coming from BWEs and a transliteration module. The combination score of source word s and translation candidate t is a simple weighted sum:

$$\text{sim}(s,t) = \gamma_{\text{bwe}}\text{sim}_{\text{bwe}}(s,t) + \gamma_{\text{tr}}\text{sim}_{\text{tr}}(s,t) \quad (1)$$

Here, $\text{sim}_{\text{bwe}}(s,t)$ is the CSLS similarity of the word pair, $\text{sim}_{\text{tr}}(s,t)$ is a similarity score based on the probability of t being the transliteration of s based on the transliteration model (see (Severini et al., 2020) for details), and the γ_i are weights of the two modules tuned on dev.

The downside of this approach is that it is strongly dependent on the order of words in the candidate lists of BWEs and transliterations. If the correct translation is contained in one of the lists but not at the first position it will not be picked as the output translation. Equation 1 has an implicit reranking capability, i.e., if a word is contained in both BWE and transliteration lists their scores get summed, which can move it higher in the final ensembled list compared to words that are present in only one

[1]Note that we use only the training portion of our dictionary for the three frequency sets (see Section 4) for this process, i.e., we split the complete training dictionary into a secondary training set and a secondary test set.

list. On the other hand, this reranking is limited as we show in our experiments. Moreover, a simple linear combination, only based on the final scores of the two modules, does not give enough flexibility to integrate other features that could help decide if a lower ranked candidate should be picked as the final translation. We aim to overcome these problems in our proposed system.

Classification Baseline The second baseline is Heyman et al. (2017)'s approach. It is a neural classifier that takes word-level information and character-level information as input. The system has two parts. The first is a character-level RNN that aims to learn orthographic similarity of word pairs. At time-step i of the input sequence, it feeds the concatenation of the i^{th} character embedding of the source and target language words to the RNN layer. The second part is the concatenation of the BWEs of the two words learned independently of the model; based on this the model aims to learn the similarity of the two semantic representations. Dense layers are applied on top of the two modules before the output softmax layer. The classifier is trained using positive and negative word pair examples. Negative examples are randomly generated for each positive one in the training lexicon. However, since characters at the same time steps are compared in the RNN module, transliteration word pairs with 1-to-many character correspondences are hard to handle correctly – any shift in the alignment then affects subsequent pairs of letters. Furthermore, apart from the source and target BWEs and their interpolated character string representation (given by the RNN module), the feed-forward layer has no information to decide whether a source word should be translated based on transliteration or based on BWEs.[2]

Classification Model Similar to Heyman et al. (2017), we employ a classification approach. Our classifier takes as input BWEs, orthography (in the form of BOEs) and additional features.

Figure 1 shows the model. It has two fully-connected feed forward layers with dropout and non-linear activation function. Its input consists of the following: BWEs of the source and target words ($\text{bwe}_s, \text{bwe}_t$), their BOEs ($\text{boe}_s, \text{boe}_t$), log frequency values (f_s, f_t) and their absolute difference, log length values (l_s, l_t), the similarity value of source and target BWEs (sim_{bwe}), the conditional probability $P(t|s)$ computed by seq2seqTr (which we call sim_{tr}) and the log of the position of the candidate in the candidate list (e.g., 1st, 2nd, etc.) for BWEs and for seq2seqTr ($\text{pos}_{\text{bwe}}, \text{pos}_{\text{tr}}$). The intuition behind f_s and f_t is that transliteration happens more often in case of rare words but corresponding word pairs should have similar frequencies, hence the feature indicating their difference. Features l_s and l_t supply further surface information about the words, while similarity and rank features are indicative of the quality of the candidates. We feed BWEs and BOEs of the word pair along with the additional features to the final feed-forward classifier.

We train the classifier to minimize the binary cross-entropy loss over positive (translation or transliteration pairs) and negative word pairs. The positive samples are the pairs in the training dictionary. We generate two negative samples for each source word: we take one candidate each from the two sorted lists of candidates (from BWEs and from seq2seqTr) at random between the 10^{th} and 20^{th} position, assuming that no positive pairs or their close synonyms belong to this range, but the words are still similar to the false candidates tested during prediction. We experimented with the range between 10 and 100, but the performance dropped since many words were used as negatives that are not realistic candidates for prediction. We note that a similar approach was developed in contemporary work (Karan et al., 2020).

3.5 Discussion

As discussed above the ensembling approach of Severini et al. (2020) has a limited reranking capability which is shown by our results. In contrast, our system is able to consider multiple candidates from the BWEs and from the transliteration candidate lists and re-rank them given the supplied information. For example, consider a non-transliteration pair such as *smoking*→курение for which a false friend exists: смокинг (*tuxedo*). The classifier can rank the false friend low since the frequencies of source and target words do not match. Although the system of Heyman et al. (2017) is based on a classifier, similar to our approach, it fails to pick the gold translation when it is not one of the top candidates in the BWEs

[2]Section 4 details minor modifications to the originally published system that make it suitable for our setup.

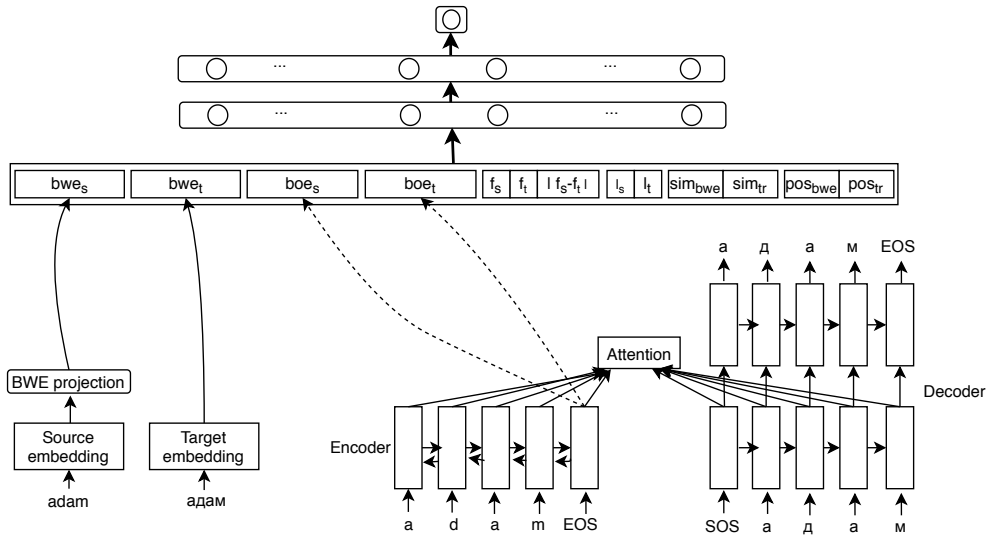

Figure 1: Classifier architecture. Top: input layer, two fully-connected layers and an output unit. Bottom right: the encoder-decoder model seq2seqTr, used as the transliteration model and to extract BOEs.

or transliteration candidates since it lacks the additional information coming from our proposed features. Statistics on the reranking capabilities of the models are shown below in section 5.1.

4 Experimental Setup

We ran our BDI experiments on the BUCC 2020 Shared Task dataset (Rapp et al., 2020); it provides both monolingual corpora and bilingual dictionaries for English-Russian. Since the official test set of the shared task is undisclosed, we relied on the released training set, a random subset of the MUSE dictionaries[3] released by Conneau et al. (2018). It is divided into three subsets, high, middle and low frequency, containing words ranked between 1-5000, 5001-20000 and 20001-40000 in the original dictionary, each having 2000 unique[4] source words. We split them into train, development and test sets (70%/15%/15%). We run experiments on the three frequency sets separately and jointly.

We followed the official setup of the BUCC shared task and relied on the WaCKy corpora (Baroni et al., 2009) as monolingual data to get the full language vocabularies and their frequencies. As MWEs we used the fasttext skipgram models (Bojanowski et al., 2017) released by the shared task organizers, which were trained using the WaCKy corpora with the following parameters: minimum word count 30; vector dimension 300; context window size 7; number of negatives sampled 10 and number of epochs 10. To align them we used VecMap (Artetxe et al., 2018) in a supervised setup using only high frequency word pairs for training, since less frequent words can be detrimental for mapping quality (Vulic and Korhonen, 2016). We use the BWEs of the 200K most frequent words following (Conneau et al., 2018). In addition, in an ablation study described below we used the 1000 word pairs of the NEWS 2010 transliteration mining test set (Kumaran et al., 2010) to test the quality of the BOEs.

For seq2seqTr we use learning rate 0.01, batch size 32, hidden size 128, single encoder and single decoder layers with the "dot" attention method of Luong et al. (2015) that compares the states with their dot product score. As mentioned above we use the same architecture for building BOEs, but we use different parameters. We tuned the hidden size (2000) of the GRU layers (used as the BOEs) and the character embedding size (300) along with the classifier on the BUCC high, middle and low frequency dev sets jointly.

For training the BDI classifier we used Adam optimizer (Kingma and Ba, 2014) with learning rate 0.001, batch size 32, 2 hidden layers on top of the described features with hidden size 300, dropout 0.01

[3]Contains up to 100K word pairs, translated with a MT system.

[4]Some words have multiple translation options.

	All	High	Mid	Low
BWEs with CSLS	0.29 (0.46)	0.47 (0.73)	0.29 (0.45)	0.11 (0.21)
Transliteration	0.05 (0.10)	0.05 (0.10)	0.06 (0.09)	0.05 (0.11)
(Severini et al., 2020)	0.33 (0.52)	0.50 (0.76)	**0.33** (0.51)	0.16 (0.30)
(Heyman et al., 2017)	0.21 (0.39)	0.28 (0.52)	0.22 (0.40)	0.14 (0.25)
(Heyman et al., 2017) w/ features	0.30 (0.48)	0.47 (0.72)	0.29 (0.45)	0.15 (0.25)
Proposed w/o features	0.22 (0.38)	0.33 (0.55)	0.23 (0.36)	0.12 (0.24)
Proposed w/o BOEs	0.31 (0.48)	0.50 (0.75)	0.30 (0.47)	0.12 (0.22)
Proposed	**0.36 (0.55)**	**0.55 (0.76)**	**0.33** (**0.56**)	**0.19 (0.33)**

Table 1: $acc@1$ ($acc@5$) on the test set for All (High+Mid+Low), High, Mid and Low. The first two lines are obtained by taking the top-1 or top-5 (in brackets) highest scoring candidates from BWE candidates or seq2seqTr candidates. For $acc@1$, Severini et al. (2020) rank $m = 100$ candidates while Heyman et al. (2017) and we rank $m = 2$ and $m = 4$, respectively. For $acc@5$, Severini et al. (2020) rank $m = 100$ candidates while Heyman et al. (2017) and we rank $m = 5$.

and ReLu activation function on the inner layer. We used early stopping on the joint dev set, decreasing the learning rate by 0.1 after not improving for 10 steps. The encoder model is kept frozen during the classifier training.

All our models are implemented in Python using PyTorch (Paszke et al., 2019), including the re-implementation of (Heyman et al., 2017). We had to modify the original setup of Heyman et al. (2017) since it relied on Levenshtein distance to look for translation candidates for source words, which is not applicable in case of language pairs with different scripts; we instead use the same transliteration candidates as in our proposed system. Furthermore, we use BWEs as the word representations as opposed to the original work where MWEs were used and the system had to learn the alignment.

5 Experiments and Results

In this section, we show the main results of our approach and compare them with the baselines. We also show an evaluation of the BOEs on the NEWS 2010 shared task (Kumaran et al., 2010) to better understand their quality on a dataset that only contains transliteration pairs.

Table 1 shows the main results for our BDI system. Our evaluation measure is $acc@n$ ($n \in \{1, 5\}$): we take n predictions from a given model and consider the source word correctly translated if any of the n predictions is the gold translation. Other than the two baseline systems we show BDI performance by taking the n highest scoring words as predictions from only BWEs or only transliteration candidates. We tuned the parameters of all systems on the joint development set, except that we followed the approach of Severini et al. (2020) and tuned ensembling weights on the three frequency sets separately. Since the systems of Severini et al. (2020), Heyman et al. (2017) and our approach are able to re-rank translation candidates, we tune the number of candidates (m) considered during prediction on the development set, i.e., we take the m highest scoring words from both BWEs and seq2seqTr lists, re-score them using one of the mentioned systems and consider the first n as translations. $m = 100$ works best on the development set for (Severini et al., 2020), $m = 2$ for (Heyman et al., 2017), $m = 1$ works best for (Heyman et al., 2017) with features, and $m = 4$ works best for our system with $acc@1$. When measuring $acc@5$, $m = 5$ works best for the classifiers and $m = 100$ for (Severini et al., 2020). Larger m values lead to worse predictions due to noisy elements in the candidate lists in case of the classifiers. In contrast, (Severini et al., 2020) is more robust against noisy elements since it is only able to rerank if a candidate word is contained in both lists. Given that the BUCC test set is divided into frequency subsets, we analyze the performance also for those.

Table 1 shows that our approach outperforms all previous approaches both on the joint ("All") and the separate frequency sets. The BWE based approach in the first row of the table achieves high performance on the high frequency set, but it suffers a significant drop as word frequency decreases. The transliteration model by itself managed to correctly induce some of the words achieving around 10% $acc@5$, which

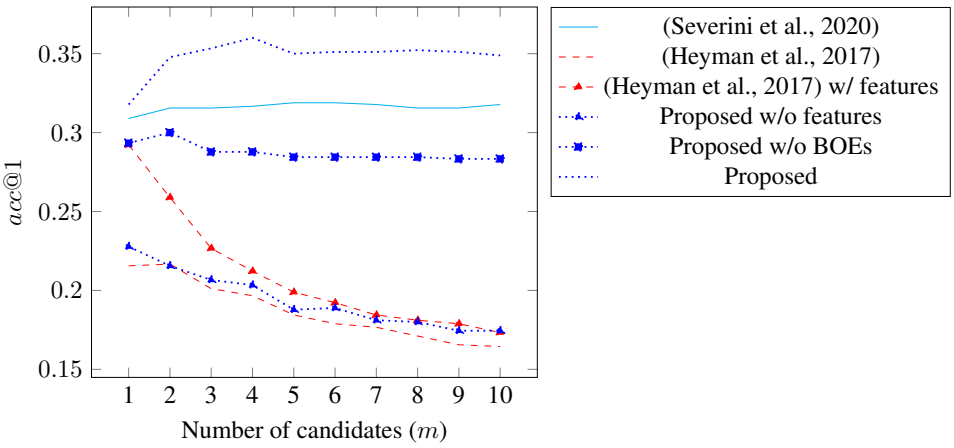

Figure 2: acc@1 on the development set as a function of the number of candidate words (e.g., 2 means 2 candidates from BWEs and 2 from seq2seqTr).

shows that a significant amount of words have to be transliterated in the datasets. The ensembling approach of Severini et al. (2020) combines the two sources of information well – the acc@1 score is almost the sum of the two combined models. It also outperforms the classifier baseline of Heyman et al. (2017). As mentioned above, best results were achieved with the classifier baseline when considering only 1 or 2 candidates each from BWEs and seq2seqTr, indicating that the system is struggling with reranking candidates. In contrast, our approach is able to exploit the additional candidates and achieves best results on all frequency sets in terms of acc@5 and three out of four sets in terms of acc@1.

5.1 Reranking Analysis

To analyze the reranking capability of our model, Figure 2 shows the performance as a function of the number of candidate words (m) on the dev set. The performance of our model improves as the number of candidates increases up to $m = 4$, and after a small performance drop at $m = 5$, it has a stable performance until 10 candidates, which further emphasizes that it is able to re-rank the candidates. When no features are used for our model, the best result is with one candidate and the performance decreases together with the re-ranking capability when using more candidates, indicating that the features are relevant to the system. Finally, we can see the behavior of our model when we rely on word embeddings and features but not on BOE information. The model acts similarly to the full version improving the results by using a few more candidates but performing near constantly after a drop as the number of candidates gets larger. On the other hand, the performance is constantly lower than the results of the full system meaning that the BOEs play a crucial role and are able to encode the orthographic structure of the words. We also added our novel features to the classifier of Heyman et al. (2017) to show the performance gain by themselves. Based on Figure 2 and Table 1, we can conclude that features clearly improve the performance of the baseline system, meaning that they are useful to decide if the BWEs or the character-level information should be emphasized. On the other hand, they are not enough to be able to positively exploit more translation candidates than 1 from this model. Similarly to our model, the performance of Severini et al. (2020) improves constantly as m increases and plateaus around $m = 100$. Still, our approach achieves better performance overall, since it does not require a given translation candidate to be contained in both candidate lists to be reranked. On the other hand, the performance of the classification based models drop significantly when $m > 10$ due to the noisier candidate lists while (Severini et al., 2020) is more robust against such noise since the noisy elements of the BWE and transliteration candidate lists are non-overlapping most of the time, thus they are not getting reranked. Our approach achieves 29.67% acc@1 with $m = 100$ but we only show $m \leq 10$ in figure 2 for simplicity.

We computed statistics, on the dev set, on the number of cases where a candidate is correctly chosen by the model and it is at rank 1 in neither the BWE nor the seq2seqTr ranking. We analyze the cases

	P	R	F
(Jiampojamarn et al., 2010)	88.0	86.9	87.5
(El-Kahky et al., 2011)	92.1	92.5	92.3
(Nabende, 2011)	-	-	82.5
(Sajjad et al., 2017)	67.1	97.1	79.4
BOEs	47.0	87.2	61.1
BOEs best	88.8	68.2	77.1

Table 2: Precision, Recall and F-measure for our BOEs and for state-of-the-art models on transliteration mining. (Sajjad et al., 2017) and our system are unsupervised while the others are (semi-) supervised.

where the best m value on the dev is greater than 1. Our model with $m = 4$ correctly predicts 15.7% of the candidates that are not in the first place while (Heyman et al., 2017) with $m = 2$ predicts only 8.8% and (Severini et al., 2020) with $m = 100$ predicts only 3.6%. Our model without BOEs is able to find only 3.8% of the candidates; thus, BOEs play a crucial role for the reordering capability of our system. We also consider the cases where the correctly chosen candidates are not at rank 1 against the total number of correct pairs found. 29.3% of the candidates correctly found by our model are not first ranked according to BWE and seq2seqTr, while 27.1% is the corresponding percentage for (Heyman et al., 2017). Note that the number of correct pairs are different for the two models and our proposed model translated more words correctly. Keeping this in mind, the small difference between the two models indicates that our model does not only have a performance advantage because of the better reranking capability, but it is also better at deciding to choose the first candidate from either the BWE or seq2seqTr lists.

5.2 BOE Evaluation

As shown above, BOEs are a crucial part of our classification model because they encode the similarity of two words based on their orthographic structure and not on their semantic meaning. The model with 2000 hidden BOE size and 300 character embedding size worked best on the BDI development set. To check the quality of the BOEs in a task where they are the only source of information, we conduct an evaluation on transliteration mining using the NEWS 2010 En-Ru test set (Kumaran et al., 2010). The task consists of the development of a mining system for identifying single word transliteration pairs from the Wikipedia Inter-Language Links (WIL) dataset in one or more language pairs. In particular, participants are required to identify word pairs in parallel sentences that are transliterations of each other. Note that the organizers provided a seed dataset of 1K transliteration pairs and a noisy training set which we ignore in these experiments and use the BUCC training dictionaries as already described to show the quality of BOEs which were used for the BDI task.

We pre-process the test sentences similar to (Kumaran et al., 2010). Given a pair of parallel sentences, for each word in the first sentence we look for the word in the second that has the highest score according to the model. We used the same model to obtain BOEs of words as in the BDI classifier. To get the score of two words, we calculate the cosine similarity of their BOEs, and we used a threshold of 0.5 to discriminate between pairs that are transliterations vs. those that are not (Sajjad et al., 2017). In Table 2 we show the precision, recall and f-measure for state-of-the-art models and our BOEs on this task. Our BOEs together with (Sajjad et al., 2017) are unsupervised and, although the BOEs are not specific for this task, they perform well. The last row of Table 2 shows the results when a threshold of 0.7 is used, that is, the best performing threshold found on the test set, thus it can be viewed as an oracle experiment. With this more accurate parameter, the system was able to reach better results compared to the naive threshold selection, which indicates the need for a development set. On the other hand, in this ablation study our goal was not to develop the best transliteration mining approach but to show the quality of BOEs. The good performance shows that BOEs have a universal embedding property of representing English and Russian words in a shared space although they use different scripts.

6 Conclusion

Bilingual Dictionary Induction is a relevant task for many applications and it is an important building block in the area of MT. In this paper we described our system for BDI for language pairs with different scripts focusing on words for which semantic information alone is insufficient. We combined semantic and orthographic information via transliteration. Our proposed model has the novel ability to make a reasonable decision on which source of information to choose via a classification approach that exploits – together with manually designed features – word embeddings and character-level information (BOEs). Our novel BOEs were learned by a language agnostic transliteration system. We tested our system on the English-Russian BUCC 2020 dataset and we showed improved results compared to the baselines. We also showed that our model is able to re-rank the candidate words better in contrast to other approaches. We evaluated our system on high, middle and low frequency sets separately and jointly. Finally, we evaluated our BOEs by running transliteration mining on the NEWS 2010 dataset, showing that they achieve good performance even if they were not meant for that specific task. Also, BOEs were shown to be able to encode the orthographic structure of words independent of the language. That means that they are universal embeddings that represent transliteration pairs similarly – a property that is useful for other downstream tasks as well. All in all, we presented a system able to combine word-level information with character-level information by means of transliteration and classification models for BDI that improved the baseline results.

Acknowledgements

We gratefully acknowledge funding for this work by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreements № 640550 and № 740516). The work was also supported by the German Research Foundation (DFG; grant FR 2829/4-1).

References

Mikel Artetxe and Holger Schwenk. 2019. Margin-based Parallel Corpus Mining with Multilingual Sentence Embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2019. An Effective Approach to Unsupervised Machine Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 194–203.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Fabienne Braune, Viktor Hangya, Tobias Eder, and Alexander Fraser. 2018. Evaluating bilingual word embeddings on the long tail. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 188–193.

Nancy Chen, Rafael E Banchs, Min Zhang, Xiangyu Duan, and Haizhou Li. 2018. Report of news 2018 named entity transliteration shared task. In *Proceedings of the seventh named entities workshop*, pages 55–73.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word Translation Without Parallel Data. In *Proceedings of the International Conference on Learning Representations*, pages 1–14.

Paula Czarnowska, Sebastian Ruder, Edouard Grave, Ryan Cotterell, and Ann Copestake. 2019. Don't Forget the Long Tail! A Comprehensive Analysis of Morphological Generalization in Bilingual Lexicon Induction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 974–983.

Ali El-Kahky, Kareem Darwish, Ahmed Saad Aldein, Mohamed Abd El-Wahab, Ahmed Hefny, and Waleed Ammar. 2011. Improved transliteration mining using graph reinforcement. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1384–1393.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL-08: Hlt*, pages 771–779.

Geert Heyman, Ivan Vulić, and Marie Francine Moens. 2017. Bilingual lexicon induction by learning to combine word-level and character-level representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1085–1095.

Ann Irvine and Chris Callison-Burch. 2017. A comprehensive analysis of bilingual lexicon induction. *Computational Linguistics*, 43(2):273–310.

Sittichai Jiampojamarn, Kenneth Dwyer, Shane Bergsma, Aditya Bhargava, Qing Dou, Mi-Young Kim, and Grzegorz Kondrak. 2010. Transliteration generation and mining with limited training resources. In *Proceedings of the 2010 Named Entities Workshop*, pages 39–47.

Mladen Karan, Ivan Vulić, Anna Korhonen, and Goran Glavaš. 2020. Classification-Based Self-Learning for Weakly Supervised Bilingual Lexicon Induction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6915–6922.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference for Learning Representations*.

Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition*, pages 9–16.

A Kumaran, Mitesh M Khapra, and Haizhou Li. 2010. Whitepaper of news 2010 shared task on transliteration mining. In *Proceedings of the 2010 Named Entities Workshop*, pages 29–38.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-Based & Neural Unsupervised Machine Translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049.

Haizhou Li, A Kumaran, Vladimir Pervouchine, and Min Zhang. 2009. Report of news 2009 machine transliteration shared task. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 1–18.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations*.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013b. Exploiting Similarities among Languages for Machine Translation. *CoRR*, abs/1309.4.

Peter Nabende. 2011. Mining transliterations from wikipedia using dynamic bayesian networks. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 385–391.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Reinhard Rapp, Pierre Zweigenbaum, and Serge Sharoff. 2020. Overview of the Forth BUCC Shared Task: Bilingual Dictionary Induction from Comparable Corpora. In *Proceedings of the 13th Workshop on Building and Using Comparable Corpora*, pages 1–6.

Parker Riley and Daniel Gildea. 2018. Orthographic Features for Bilingual Lexicon Induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 390–394.

Hassan Sajjad, Helmut Schmid, Alexander Fraser, and Hinrich Schütze. 2017. Statistical models for unsupervised, semi-supervised, and supervised transliteration mining. *Computational Linguistics*, 43(2):349–375.

Silvia Severini, Viktor Hangya, Alexander Fraser, and Hinrich Schütze. 2020. LMU Bilingual Dictionary Induction System with Word Surface Similarity Scores for BUCC 2020. In *Proceedings ofthe 13th Workshop on Building and Using Comparable Corpora*, pages 49–55.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Ivan Vulic and Anna Korhonen. 2016. On the Role of Seed Lexicons in Learning Bilingual Word Embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 247–257.